NAME: Daniel Gonzalez Cedre NETID: —

Exam 1.p.fa.2024
Principles of Computing
30th of October, 2024

UNIVERSITY OF NOTRE DAME

1 *Answer the following questions either True or False.*

The first five questions on this page are about the *shell*; the rest are about *Python*.

i. The *root* of a UNIX-based file system is a directory named `/root`.

- ○ *True*
- ⊗ *False*

this would be a directory named "root" located inside of /. the root of the filesystem is the directory named "/"; the first directory.

ii. The `cd` command can be used to change a directory's name.

- ○ *True*
- ⊗ *False*

"cd" is the command for changing the directory you are in; you use it to navigate through the file system. "mv" is the command for moving a file from one place to another, and this can be used to rename files.

iii. The name `~` is an alias for the current user's *"home"* directory.

- ⊗ *True*
- ○ *False*

iv. `mv(pumpkins.py, ~/patch)` will move `pumpkins.py` to the `patch` directory in the user's home folder.

- ○ *True*
- ⊗ *False*

this is using Python function syntax, which doesn't work to invoke bash/zsh commands in the terminal. It should say `mv pumpkins.py ~/patch/`

v. `touch closet/skeletons.csv` will create a new file named `skeletons.csv` within `closet` in the home folder.

- ○ *True*
- ⊗ *False*

↑ assuming "closet" is a directory that exists in the current directory, this will create skeletons.csv inside of closet in the current directory. It

vi. `", ".join(["this is Halloween", "this is Halloween"])[19::1]` is valid syntax.

- ○ *True*
- ○ *False*

Should read `touch ~/closet/skele-`

now, it will make skeletons.csv inside of ~/closet.

vii. Every value in *Python* is an object, and every object has a type.

- ⊗ *True*
- ○ *False*

viii. Every method is a function in *Python*.

- ⊗ *True*
- ○ *False*

ix. Every function returns a value.

- ⊗ *True*
- ○ *False*

x. `print("Hello, world!")` returns an object of type `str`.

- ○ *True*
- ⊗ *False*

print(message) returns None, the special NoneType object, because its not supposed to return anything meaningful.

NAME: NETID:

2 *Answer the following questions without justification.*

i. What is the value of `scary_string` at the end of the following block?

```
scary_string = "i am the one hiding under your bed".split(" ")
scary_string[0:2] += ["not"]
scary_string = " ".join(scary_string)
```

"i am not the one hiding under your bed"

ii. What is the result of the following expression?

```
len("".join("everyone hail to the pumpkin song\n".split(" ")))
```

even though it appears as \n, this is rendered as one newline character; this is an escape sequence.

this has the effect of removing the spaces.

there are 28 alpha-numeric characters in the string, 5 space characters, and 1 newline character. the .split(" ") separates the string by spaces, and the "".join(...) recombines them separated by the empty string.

29.

iii. What is the value of `spooky_func(" ")` with the definition below?

```
def spooky_func(ghost):
  x = "i am the shadow on the moon at night".split(ghost)
  return x.insert(2, "not")
```

The list method .insert(.,.) does not return a meaningful value because it directly modifies the list it is called from, which is possible because lists are mutable; so, the method returns None instead.

None.

iv. How many times will line 4 be evaluated in the block of code below?

```
def rattle(something):
  return len(something) % 2 == 0
bones = list(range(101))
while rattle(bones):
  bones = bones[:bones[-1]/2]
```

len(bones) = 101, so rattle(bones) returns False.

once.

v. What value is returned as a result of calling the function below?

```
def trick_or_treat():
  ll = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
  fear = 0
  for x in ll:
    for y, z in enumerate(x):
      if z % (y + 1) == 0:
        fear += z
    if fear == 13:
      return "trick"
  return "treat"
```

"treat"

This is an exercise in walking through code by-hand.
If you have trouble with this one, make sure you walk through the different iterations of these "for" loops and keep track of the values of x, y, z, and fear with pen & paper.

NAME: NETID:

3 *Answer the following questions without justification.*

i. What *type of error* will be thrown when this code is run?[1]

[1] You may assume the user provides valid, sensible input (*e.g.*, 10) when prompted.

```
def jump_scare(days_left):
  return days_left == 7
upper_bound = input("Enter your maximum tolerance for terror: ")
for i in range(upper_bound):
  print(jump_scare(days_left))
```

the input(...) function returns a [str] value, but the range(...) function requires its input(s) to be [int] valued.

TypeError

ii. Indicate *all lines* in the following code block that would *throw an error.*

```
dungeon = []
for i in range(5):
  room = []
  for j in range(5):
    room += [i*j]
  dungeon.append(room)
if sum(dungeon) % 13 == 0:
  return "creepy"
return "wet"
```

dungeon is a list of lists, and the sum(.) function only performs addition, not concatenation.

we are not inside of a [def] block for defining a function.

7, 8, 9

iii. The block of code below throws an error; on *what line* does it occur?

```
from random import randint
weight = "0"
giles_corey = "185"
while plea not in ["guilty", "not guilty"]:
  weight += int(f"{randint(1, 10)}")
  giles_corey -= int(weight)
  plea = "more weight"
```

giles_corey is a [str] object, and we are trying to evaluate giles_corey = giles_corey - int(weight). this causes a TypeError.

6

iv. What is `monster_sort([3, 1, -1, 0, 5, 3])` based on the code below?

```
def monster_sort(ll):
  rr = []
  for l in ll:
    scared = False
    for i, r in enumerate(rr):
      if (l < r) and (not scared):
        rr.insert(i, l)
        scared = True
    if not scared:
      rr.append(l)
  return rr
```

[-1, 0, 1, 3, 3, 5]

this is just another implementation of insertion sort.

NAME: _____ NETID: _____

4 *Program a solution to the following question in Python.*

Given two vectors $x = [x_0, x_1, \ldots, x_{n-1}]$ and $y = [y_0, y_1, \ldots, y_{n-1}]$ of the same positive integer length n, we compute the *inner product* of x with y, which we denote $\langle x, y \rangle$, as follows.

$$\langle x, y \rangle = x_0 y_0 + x_1 y_1 + \cdots + x_{n-1} y_{n-1}$$

For example, the inner product of `[1,2,3]` and `[2,4,8]` is below.

$$\begin{aligned} \langle [1,2,3], [2,4,8] \rangle &= 1 \cdot 2 + 2 \cdot 4 + 3 \cdot 8 \\ &= 2 + 8 + 24 \\ &= 34 \end{aligned}$$

Write a function in *Python* that takes two lists of floating point numbers representing *vectors* as inputs and returns their *inner product* as output *if they are the same length*, but returns the string `"ERROR"` otherwise.

RESTRICTIONS:

- No use of `import` statements.
- No use of methods for any type.
- No use of the built-in functions:
  - `sorted()`
  - `sum()`
  - `max()`
  - `min()`
  - `abs()`
- No use of functions, methods, types, or control structures that we have not yet covered in class nor problem sets.

NOTE: you may name your function and your variables whatever you'd like.

```
def inner_product(x, y):
    result = 0
    [illegible]
        [illegible]
    [illegible]
    if len(x) != len(y):
        return "ERROR"
    for i in range(len(x)):
        result += x[i] * y[i]
    return result
```